RPC BROKER
TECHNICAL MANUAL

Version 1.1

Revised May 2002

Department of Veterans Affairs
V*IST***A** System Design & Development (SD&D)
Information Infrastructure Service (IIS)

Document Revision History

The following table displays the revision history for this document. Revisions to the documentation are based on patches and new versions released to the field.

Date	Revision	Description	Author
05/08/02	3.0	Revised Version for Patch 26.	Thom Blom, Oakland OIFO
04/08/02	2.0	Revised Version for Patch 13.	Thom Blom, Oakland OIFO
09/97	1.0	Initial RPC Broker Version 1.1 software release.	Thom Blom, San Francisco OIFO

ℹ For a complete list of patches released with the RPC Broker V. 1.1 software, please refer to "Appendix A—Patch Revision History."

Contents

Figures

Orientation

How to Use this Manual

Throughout this manual, advice and instructions are offered regarding the use of the RPC Broker V. 1.1 and the functionality it provides for Veterans Health Information Systems and Technology Architecture (V*IST*A) and commercial off-the-shelf (COTS) software products.

There are no special legal requirements involved in the use of the RPC Broker's Interface.

This manual uses several methods to highlight different aspects of the material:

- Various symbols are used throughout the documentation to alert the reader to special information. The following table gives a description of each of these symbols:

Symbol	Description
ℹ	Used to inform the reader of general information including references to additional reading material
⚠	Used to caution the reader to take special notice of critical information

Table 1: Documentation symbol descriptions

- Descriptive text is presented in a proportional font (as represented by this font).
- "Snapshots" of computer online displays (i.e., roll-and-scroll screen captures/dialogs) and computer source code are shown in a *non*-proportional font and enclosed within a box. Also included are Graphical User Interface (GUI) Microsoft Windows images (i.e., dialogs or forms).
    - User's responses to online prompts will be boldface type.
    - The "**<Enter>**" found within these snapshots indicate that the user should press the Enter or Return key on their keyboard.
    - Author's comments are displayed in italics or as "callout" boxes.

    ℹ Callout boxes refer to labels or descriptions usually enclosed within a box, which point to specific areas of a displayed image.

- Object Pascal code uses a combination of upper- and lowercase characters. All Object Pascal reserved words are in boldface type.
- All uppercase is reserved for the representation of M code, variable names, or the formal name of options, field and file names, and security keys (e.g., the XUPROGMODE key).

Commonly Used Terms

The following is a list of terms and their descriptions that you may find helpful while reading the RPC Broker documentation:

Term	Description
Client	A single term used interchangeably to refer to a user, the workstation (i.e., PC), and the portion of the program that runs on the workstation.
Component	A software object that contains data and code. A component may or may not be visible. For a more detailed description, see the "Borland Delphi for Windows User Guide."
GUI	The Graphical User Interface application that is developed for the client workstation.
Host	The term Host is used interchangeably with the term Server.
Server	The computer where the data and the RPC Broker remote procedure calls (RPCs) reside.

Table 2: Commonly used RPC Broker terms

Please refer to the "Glossary" for additional terms and definitions.

How to Obtain Technical Information Online

Exported file, routine, and global documentation can be generated through the use of Kernel, MailMan, and VA FileMan utilities.

Methods of obtaining specific technical information online will be indicated where applicable under the appropriate topic.

Help at Prompts

V*IST*A software has online help and commonly used system default prompts. In roll-and-scroll mode users are strongly encouraged to enter question marks at any response prompt. At the end of the help display, you are immediately returned to the point from which you started. This is an easy way to learn about any aspect of **V*IST*A** software.

To retrieve online documentation in the form of Help in **V*IST*A** roll-and-scroll software:

- Enter a single question mark ("**?**") at a field/prompt to obtain a brief description. If a field is a pointer, entering one question mark ("**?**") displays the HELP PROMPT field contents and a list of choices, if the list is short. If the list is long, the user will be asked if the entire list should be displayed. A YES response will invoke the display. The display can be given a starting point by prefacing the starting point with an up-arrow ("^") as a response. For example, **^M** would start an alphabetic listing at the letter M instead of the letter A while **^127** would start any listing at the 127th entry.
- Enter two question marks ("**??**") at a field/prompt for a more detailed description. Also, if a field is a pointer, entering two question marks displays the HELP PROMPT field contents and the list of choices.
- Enter three question marks ("**???**") at a field/prompt to invoke any additional Help text that may be stored in Help Frames.

Obtaining Data Dictionary Listings

Technical information about files and the fields in files is stored in data dictionaries. You can use the List File Attributes option on the Data Dictionary Utilities submenu in VA FileMan to print formatted data dictionaries.

For details about obtaining data dictionaries and about the formats available, please refer to the "List File Attributes" chapter in the "File Management" section of the "VA FileMan Advanced User Manual."

Assumptions About the Reader

This manual is written with the assumption that the reader is familiar with the following:

- **V*IST*A** computing environment (e.g., Kernel Installation and Distribution System [KIDS])
- VA FileMan data structures and terminology
- Microsoft Windows
- M programming language

No attempt is made to explain how the overall **V*IST*A** programming system is integrated and maintained. Such methods and procedures are documented elsewhere. We suggest you look at the various VA home pages on the World Wide Web for a general orientation to **V*IST*A**. For example, go to the System Design & Development (SD&D) Home Page at the following web address:

http://vista.med.va.gov/

This manual does provide, however, an explanation of the RPC Broker, describing how it can be used in a client/server environment.

Reference Materials

Readers who wish to learn more about the RPC Broker should consult the following:

- "RPC Broker Getting Started with the Broker Development Kit (BDK)" (written for programmers)
- "RPC Broker Developer's Guide" (i.e., BROKER.HLP, online help designed for programmers, distributed in the BDK)
- "RPC Broker Systems Manual"
- "RPC Broker Installation Guide"
- "RPC Broker Release Notes"
- The "MIRMO/ISC Operations Document," Chapter 10
- "Programming Standards and Conventions (SAC)"
- RPC Broker Home Page at the following web address:

  http://vista.med.va.gov/broker/

  This site provides announcements, additional information (e.g., Frequently Asked Questions [FAQs], advisories), documentation links, archives of older documentation and software downloads.

Broker documentation is made available online, on paper, and in Adobe Acrobat Portable Document Format (.PDF). The .PDF documents must be read using the Adobe Acrobat Reader (i.e., ACROREAD.EXE), which is freely distributed by Adobe Systems Incorporated at the following web address:

http://www.adobe.com/

For more information on the use of the Adobe Acrobat Reader, please refer to the "Adobe Acrobat Quick Guide" at the following web address:

http://vista.med.va.gov/iis/acrobat/index.html

DISCLAIMER: The appearance of external hyperlink references in this manual does not constitute endorsement by the Department of Veterans Health Administration (VHA) of this Web site or the information, products, or services contained therein. The VHA does not exercise any editorial control over the information you may find at these locations. Such links are provided and are consistent with the stated purpose of this VHA Intranet Service.

1. Introduction

The RPC Broker Technical Manual" provides descriptive information and instructions on the use of the Remote Procedure Call (RPC) Broker (also referred to as "Broker") software within the VA's Veterans Health Information Systems and Technology Architecture (**V***IST***A**) environment. This document is intended for systems managers—Information Resource Management (IRM) personnel who are responsible for implementing and maintaining this software, application programmers, and developers. It acquaints system managers with the utilities, software structure, and functionality of the RPC Broker system modules, including information about the routines and files that comprise this software. It also has information about the software's structure and recommendations regarding its efficient use (e.g., routine mapping). Additional information on installation, security, management features, and other requirements is also included.

Product Overview

The RPC Broker is considered to be part of the infrastructure of **V***IST***A**. It establishes a common and consistent foundation for communication between clients and **V***IST***A** M servers.

The RPC Broker is a bridge connecting the client application front-end on the workstation (e.g., Delphi GUI applications) to the M-based data and business rules on the server. It links one part of a program running on a workstation to its counterpart on the server. The client and the server can be, and most often are, written in different computer languages. Therefore, the RPC Broker bridges the gap between the traditionally proprietary **V***IST***A** and COTS/HOST products.

The RPC Broker includes:

- A common communications driver for the M server interface that handles the device-specific characteristics of the supported communications protocol.
- An interface component on the M server, separate from the communications driver, that interprets client messages, executes the required code, and eventually returns data to the communications driver.
- A common file on the M server that all applications use to store the information about the queries to which they respond (i.e., REMOTE PROCEDURE file [#8994]).
- The Client Agent application that runs on client workstations, supporting single signon.
- The TRPCBroker component for Delphi, enabling development of client applications that can communicate via the RPC Broker.
- A dynamic link library (DLL) that provides access to RPC Broker functionality for development environments other than Delphi.

2. Implementation and Maintenance

The "RPC Broker Installation Guide" provides detailed information regarding the installation of the RPC Broker. It also contains many requirements and recommendation regarding how the Broker should be configured. Be sure to read the Installation Guide before attempting to install the RPC Broker.

Site Parameters

The following two areas of the Broker require site parameter review and configuration:

Functional Area	For Information, Please See…
Broker Listeners	"RPC Broker Systems Manual," "RPC Broker Site Parameters File" section
Single Signon	"RPC Broker Systems Manual," Integrated Auto Signon For Multiple Users" section

Table 3: RPC Broker functionality requiring site parameter review

Performance and Scalability

Current performance statistics are limited. However, results indicate that the processing time and resources consumed by the Broker itself are minimal. The RPC Broker doesn't introduce any additional overhead to the messages sent between the client and the server.

The RPC Broker listener does not tend to get overloaded, because it jobs off incoming requests to another process and then keeps listening for another request. This action is only limited by the number of partitions the M configuration supports.

Performance should instead be measured at the application level to determine the amount of resources consumed by **V***IST***A** client/server applications that use the Broker. Performance and scalability, from a site's point of view, have been impacted by the load introduced by application executing on the host system, as opposed to the load introduced by the RPC Broker itself.

3. File List

M Server Files

The RPC Broker consists of a single global with two files. This chapter describes the RPC Broker files including the file number, file name, global location, and description of the files.

File #	File Name	Global Location
8994	REMOTE PROCEDURE	^XWB(8994,
8994.1	RPC BROKER SITE PARAMETERS	^XWB(8994.1,

Table 4: RPC Broker files

REMOTE PROCEDURE File (#8994)

This file is used as a repository of server-based procedures (i.e., remote procedure calls [RPCs]) in the context of the Client/Server architecture. All RPCs used by any site-specific client/server application software using the RPC Broker interface must be registered and stored in this file. Applications running on client workstations can invoke (call) the RPCs in this file to be executed by the server and the results are returned to the client application. Each RPC is associated with an entry point (i.e., ROUTINE with optional TAG).

The RPC subfield (#19.05) of the OPTION File (#19) points to RPC field (#.01) of the REMOTE PROCEDURE file (#8994).

Data is not distributed with this file. RPCs are distributed and installed as separate components during the installation of the RPC Broker, however.

RPC BROKER SITE PARAMETERS file (#8994.1)

Site managers can use this file to configure and adjust many characteristics of an RPC Broker installation at a site.

Data is not distributed with this file.

Client Files

End-User Workstation

..\Program Files\VISTA\Broker

Clagent.exe

Clagent.hlp

RPCTest.exe

RPCTest.hlp

..\Windows\System(32)

Bapi32.dll

XWB_R40.bpl

XWB_R50.bpl

XWB_R60.bpl

Programmer Workstation

Files installed vary depending on BDK patch level, installation choices, and Delphi version. In general, files are placed in the following directories:

..\Program Files\VISTA\BDK32\D2 (*no longer supported*)

..\Program Files\VISTA\BDK32\D3 (*no longer supported*)

..\Program Files\VISTA\BDK32\D4

..\Program Files\VISTA\BDK32\D5

..\Program Files\VISTA\BDK32\D6

..\Program Files\VISTA\BDK32\Help

..\Program Files\VISTA\BDK32\Samples\BrokerEx

..\Program Files\VISTA\BDK32\Samples\SilentSignOn (p13)

..\Program Files\VISTA\BDK32\Samples\Vb5Egcho

..\Program Files\VISTA\BDK32\Source

On Programmer Workstations, if you are recompiling the BDK32, you will probably want to remove the .dpl files from the System(32) directory. These files are in the ($Delphi)\Projects\Bpl directory in D5 and D6 and the ($Delphi)\Bin and ($Delphi)\Lib directory in D4. The files in the System32 directory are usually earlier in the path, and therefore, interfere with newly compiled packages.

4. Global Translation, Journaling, and Protection

Translation

Translation is recommended for the sole RPC Broker global (i.e., ^XWB global). The ^XWB global has the potential to be read-intensive as more and more remote procedures are added to it in the future.

For DSM and Caché Systems:

It is best to translate the global to a volume set other than ROU. In order for translation to take effect on DSM and Caché systems, DSM and Caché must be rebooted.

Cookbook recommendations should also be consulted for suggestions regarding journaling, translation, and replication; the information here may not apply.

Journaling

Journaling of this global is not required, since the ^XWB global, for the most part is static (except during the addition of new remote procedures).

Protection

The following global protection should be set:

Global Name	Protection	
	DSM for OpenVMS	Caché
^XWB	System: RWD World: RW Group: RW User: RW	Owner: RWD Group: N World: N Network: RWD

Table 5: RPC Broker Global Information

5. Routine List

This chapter contains a list of the routines exported with the RPC Broker. A brief description of the routines is provided.

Routine	**Description**
XWB2HL7	This routine contains various functions and procedures that are used by the Broker for Remote Data Views (RDV).
XWB2HL7A	This routine contains various functions and procedures that are used by the Broker for Remote Data Views (RDV).
XWB2HL7B	This routine contains various functions and procedures that are used by the Broker for Remote Data Views (RDV).
XWBBRK	This routine contains calls that are designed to parse the various attributes of the Broker messages. All of this information is used internally. In the case of large arrays sent by the client, the function BREAD is used to read in the variable length subscripts and values.
XWBBRK2	This routine is a continuation of XWBBRK. The main entry point (i.e., CAPI) actually calls the application RPC.
XWBCAGNT	Server code for RPC Broker client agent application.
XWBDRPC	This routine contains various functions and procedures that are used for deferred RPCs by the Broker for Remote Data Views (RDV).
XWBEXMPL	This routine is used to support the Broker Example application. The Broker Example application is used to test the RPC Broker connectivity, actions, and RPCs. It is distributed with the Broker.
XWBFM	This routine contains entry points used to interface to the VA FileMan database server.
XWBLIB	This routine contains various functions and procedures used by the Broker. It is best described as a library or depository.
XWBSEC	This routine contains various functions and procedures used by the Broker. Calls in this routine are used for client/server security.
XWBTCP	This routine contains functions and procedures used to control the Broker TCP/IP Listener process. Systems personnel can use calls in this routine to start, stop, and debug the Broker process.
XWBTCPC	This job is started for each Broker request. The Listener process (i.e., XWBTCPL) will receive a connection request from a client and then dispatch, using the M JOB command, XWBTCPC to manage the rest of the interaction.

Routine	Description
XWBTCPL	This is the Broker Listener process. IRM starts this job. It remains running on a system listening for TCP/IP connection requests. Once a request is received, this routine will start a separate process to manage the rest of the connection, then returns to "listening" for a new request.
XWBZ1	This routine supports the RPC Broker V. 1.0 Echo application, which was used to test RPC Broker connectivity, actions, and APIs. Note: The Echo client application is not distributed in V. 1.1 of the RPC Broker; it is replaced by the RPCTest application.

Routine Mapping

RPC Broker routines are *not* required to be mapped to any account.

6. Exported Options

The following options are exported with the RPC Broker:

Name	Menu Text	Type
XWB BROKER EXAMPLE	RPC BROKER PROGRAMMING EXAMPLE	Broker (Client/Server)
XWB LISTENER STARTER	Start All RPC Broker Listeners	Run Routine
XWB RPC TEST	RPC	Broker (Client/Server)

Table 6: Exported Options

Client/server applications are a new type of option (i.e., Type "B", Broker client/server options) in the OPTION file (#19). The user must have the client/server application option assigned to them as with any other assigned option in **V*IST*A**. The client/server application will only run for those users who are allowed to activate it.

ℹ The client/server application options will not be displayed in the user's menu tree.

XWB BROKER EXAMPLE

This option supports the Broker Example demonstration program provided in the Broker Development Kit (BDK). Developers should assign this option to themselves, if they want to try out the Broker Example application. For programmers who have the XUPROGMODE key, however, assigning this option to themselves is not necessary.

XWB LISTENER STARTER

Modified by patch XWB*1.1*9. For information on this option, please refer to the "RPC Broker Systems Manual."

XWB RPC TEST

It is recommended that the XWB RPC TEST option be given to users running Broker-based **V*IST*A** client/server applications. The RPCTEST.EXE program on the client workstation runs the RPC Broker Diagnostic Program. This tool can be used to verify and test the Broker client/server connection and signon process. It displays information about the client and the server and can be a useful debugging tool for IRM.

To enable remote troubleshooting by IRM for all users, you can put this option on the Common menu (i.e., System Command Options menu [XUCOMMAND]). This enables any user to run the RPCTEST.EXE program on their workstation at your request.

Exported RPCs

The RPC Broker distributes the following remote procedure calls (RPCs):

XWB ARE RPCS AVAILABLE
XWB CREATE CONTEXT
XWB DEFERRED CLEAR
XWB DEFERRED CLEAR ALL
XWB DEFERRED GETDATA
XWB DEFERRED RPC
XWB DEFERRED STATUS
XWB DIRECT RPC
XWB EGCHO BIG LIST
XWB EGCHO LIST
XWB EGCHO MEMO
XWB EGCHO SORT LIST
XWB EGCHO STRING
XWB EXAMPLE ECHO STRING
XWB EXAMPLE GET LIST
XWB EXAMPLE SORT NUMBERS
XWB EXAMPLE WPTEXT
XWB GET BROKER INFO
XWB GET VARIABLE VALUE
XWB FILE LIST
XWB FILENAME CHECK
XWB IM HERE
XWB IS RPC AVAILABLE
XWB REMOTE CLEAR
XWB REMOTE GETDATA
XWB REMOTE RPC
XWB REMOTE STATUS CHECK
XWB RPC LIST

7. Archiving and Purging

Archiving

There are no package-specific archiving procedures or recommendations for the RPC Broker ^XWB global or the REMOTE PROCEDURE and RPC BROKER SITE PARAMETERS files.

Purging

There are no package-specific purging procedures or recommendations for the RPC Broker ^XWB global or the REMOTE PROCEDURE and RPC BROKER SITE PARAMETERS files.

8. Callable Routines

The RPC Broker does *not* provide any callable M routines. However, other programming interfaces are provided (e.g., Delphi components, DLL, Pascal functions, and RPCs).

For information on these other programming interfaces, please refer to the "External Interfaces" chapter in this manual.

9. External Interfaces

The following external interfaces to RPC Broker functionality are provided:

RPC Broker Components

The TRPCBroker and TSharedRPCBroker components provide all functionality needed for client applications to communicate with **VISTA** M servers via the RPC Broker. The TRPCBroker and TSharedRPCBroker components are compatible with Borland Delphi V. 4.0 and greater.

Delphi V.2 and V. 3 are no longer supported. Delphi V.2 was supported prior to patch XWB*1.1*4 and Delphi V. 3 was supported prior to XWB*1.1*13.

This statement defines the extent of support relative to use of Delphi. The Office of Information (OI) will support the Broker Development Kit (BDK) running in the currently offered version of Delphi and the immediately previous version of Delphi. This level of support became effective 06/12/2000.

Sites may continue to use outdated versions of the RPC Broker Development Kit but do so with the understanding that support will *not* be available and that continued use of outdated versions will *not* afford features that may be essential to effective client/server operations in the VISTA environment. An archive of old (no longer supported) Broker Development Kits will be maintained at:

http://vista.med.va.gov/broker/archives/index.html.

For more information on the Broker components, please refer to the "RPC Broker Getting Started with the Broker Development Kit (BDK)" manual and "RPC Broker Developer's Guide" (BROKER.HLP, online help in the BDK).

TRPCBroker Dynamic Link Library (DLL)

The TRPCBroker DLL (BAPI32.DLL) provides access to RPC Broker functionality for development environments other than Delphi.

For more information on the TRPCBroker DLL, please refer to the "RPC Broker Getting Started with the Broker Development Kit (BDK)" manual and "RPC Broker Developer's Guide" (BROKER.HLP, online help in the BDK).

Pascal Functions

The following Pascal functions are provided by the TRPCBroker component:

- GetServerInfo function
- Splash Screen functions: SplashOpen and SplashClose
- Piece function
- Translate function
- Encryption functions: Decrypt and Encrypt

For more information on these Pascal functions, please refer to the "RPC Broker Getting Started with the Broker Development Kit (BDK)" manual and "RPC Broker Developer's Guide" (BROKER.HLP, online help in the BDK).

RPC Broker Remote Procedures

The following RPC is provided for use by developers:

- XWB GET VARIABLE VALUE

For more information, please refer to the "RPC Broker Getting Started with the Broker Development Kit (BDK)" manual and "RPC Broker Developer's Guide" (BROKER.HLP, online help in the BDK).

10. External Relations

Relationship to Other Packages

The RPC Broker software has been developed to aid the **V*IST*A** development community and Information Resources Management (IRM) and is considered to be part of the infrastructure of **V*IST*A**. Other infrastructure products include VA FileMan, Kernel, and MailMan. The RPC Broker will be used by all client applications written as part of **V*IST*A**. The RPC Broker fully integrates with VA FileMan V. 22.0 and Kernel V. 8.0.

The absence of RPC Broker software on an M server will disable the functioning of any client application that depends on the RPC Broker to communicate with the M Server.

It is possible that the use of RPCs will also be extended to non-client applications. In this case, the REMOTE PROCEDURE FILE must be present for those applications to function correctly.

Relationship with Kernel and VA FileMan

Before installing the RPC Broker, Kernel V. 8.0, Kernel Toolkit V. 7.3, and VA FileMan V. 22.0 must be in place and fully patched.

Relationships with Operating Systems

On the client side, it was decided that the 32-bit Microsoft Windows environment would be the supported platform. Thus, the client portions of the RPC Broker are compatible with Microsoft Windows 95 or higher, and Microsoft Windows NT 3.5 or higher.

On the server side, the RPC Broker supports the following ANSI M environments:

- Digital Standard M (DSM) V6.3-031 for OpenVMS AXP or greater
- InterSystems Caché for NT and OpenVMS

DBA Approvals and Database Integration Agreements (DBIAS)

To obtain the current list of DBIAs that the RPC Broker is a custodian of:

1. Sign on to the Forum system (forum.va.gov).
2. Go to the DBA menu.
3. Select the Integration Agreements menu.
4. Select the Custodial Package menu.
5. Choose the ACTIVE by Custodial Package option.
6. When this option prompts you for a package, enter RPC BROKER.
7. All current DBIAs for which the RPC Broker package is custodian are listed.

To obtain detailed information on a specific integration agreement:

1. Sign on to the Forum system (forum.va.gov).
2. Go to the DBA menu.
3. Select the Integration Agreements menu.
4. Select the Inquire option.
5. When prompted for "INTEGRATION REFERENCES", enter the integration agreement number of the DBIA you would like to display.
6. The option then lists the full text of the DBIA you requested.

To obtain the current list of DBIAs that the RPC Broker is a subscriber to:

1. Sign on to the Forum system (forum.va.gov).
2. Go to the DBA menu.
3. Select the Integration Agreements menu.
4. Select the Subscriber Package menu.
5. Choose the Print ACTIVE by Subscribing Package option.
6. When prompted "START WITH SUBSCRIBING PACKAGE", enter RPC BROKER (in uppercase). When prompted "GO TO SUBSCRIBING PACKAGE", enter RPC BROKER (in uppercase).
7. All current DBIAs to which the RPC Broker package is a subscriber are listed.

11. Internal Relations

No options in the RPC Broker product assume that the entry/exit logic of another option has already occurred.

12. Package-wide Variables

The RPC Broker does *not* create any package-wide variables that have received Programming Standards and Conventions Committee (SACC) exemptions.

13. Software Product Security

Security Management

There are no special legal requirements involved in the use of the RPC Broker product.

Mail Groups and Alerts

The RPC Broker does not make use of mail groups or alerts.

Remote Systems

Connections

The M server process of the RPC Broker allows connections from client applications. Connection by those client applications is subject to authentication as any normal logon requires. Client applications can use any remote procedure call (RPC) authorized to the application, if the application is authorized to the signed-on user. Data is typically exchanged between clients and the RPC Broker server. Clients can be anywhere on VA's TCP/IP network.

Encryption is used when a user's Access and Verify codes are sent from the client to the server.

In addition, an encryption API is provided for developer use in their own applications to encode and decode messages passed between client and server.

Security with the RPC Broker is a four-part process:

1. Client workstations must send a valid connection request to the M Server.
2. Users must have valid Access and Verify codes.
3. Users must be valid users of a **V***IST***A** client/server application.
4. Any remote procedure call must be registered and valid for the application being executed.

For more information regarding Broker security, please refer to Chapter 2, "Security," in the "RPC Broker Systems Manual."

Remote Data Views

The RPC Broker can be used to facilitate invocation of Remote Procedure Calls on a remote server. Applications can use either XWB DIRECT RPC or XWB REMOTE RPC to pass:

- The desired remote server
- The desired remote RPC
- Any parameters for the remote RPC

The RPC Broker on the local server uses **V***IST***A** HL7 as a vehicle to pass the remote RPC name and parameters to the remote server. **V***IST***A** HL7 is used to send any results from the remote server back to the local server. The RPC Broker on the local server then passes the results back to the client application.

ℹ WB DIRECT RPC and XWB REMOTE RPC are available only on a controlled subscription basis.

Interfacing

No *non*-VA products are embedded in or required by the RPC Broker, other than those provided by the underlying operating systems.

Electronic Signatures

Electronic signatures are not used within the RPC Broker.

Security Keys

There are *no* specific security keys exported with the RPC Broker software. However, to bypass security for development purposes, we recommend client/server application developers be assigned the XUPROGMODE security key.

All users assigned the XUPROGMODE security key can do the following:

- Run any **V***IST***A** client/server application regardless of whether it is in their menu tree or not, and
- Access any RPC without regard to the application context.

File Security

The RPC Broker establishes the following security over its files:

Number	Name	DD	RD	WR	DEL	LAYGO	AUDIT
8994	REMOTE PROCEDURE	@	@	@	@	@	@
8994.1	RPC BROKER SITE PARAMETERS	@	@	@	@	@	@

Table 7: File Security

Official Policies

Modification of any part of the RPC Broker software is strongly discouraged.

Distribution of the RPC Broker software is unrestricted.

The VHA IT Architecture Statement of Direction for FY98 prescribes *"Use of Kernel Broker for client-server communication..."*

Glossary

.001 FIELD

A field containing the internal entry number of the record.

.01 FIELD

The one field that must be present for every file and file entry. It is also called the NAME field. At a file's creation the .01 field is given the label NAME. This label can be changed.

ACCESS CODE

A code that, along with the Verify code, allows the computer to identify you as a user authorized to gain access to the computer. Your code is greater than 6 and less than 20 characters long; can be numeric, alphabetic, or a combination of both; and is usually assigned by a site manager or application coordinator. It is used by the Kernel's Sign-on/Security system to identify the user (see Verify Code).

ALERTS

Brief online notices that are issued to users as they complete a cycle through the menu system. Alerts are designed to provide interactive notification of pending computing activities, such as the need to reorder supplies or review a patient’s clinical test results. Along with the alert message is an indication that the View Alerts common option should be chosen to take further action.

ANSI MUMPS

The MUMPS programming language is a standard recognized by the **A**merican **N**ational **S**tandard **I**nstitute (ANSI). MUMPS stands for **M**assachusetts **U**tility **M**ulti-**p**rogramming **S**ystem and is abbreviated as M.

APPLICATION PACKAGE

Software and documentation that support the automation of a service, such as Laboratory or Pharmacy within VA medical centers. The Kernel application package is like an operating system relative to other **V**_IST_**A** applications.

APPLICATION PROGRAMMING INTERFACE (API)

Programmer calls provided by **V**_IST_**A** software for use by application programmers. APIs allow programmers to carry out standard computing activities without needing to duplicate utilities in their own packages. APIs also further DBA goals of system integration by channeling activities, such as adding new users, through a limited number of callable entry points.

ARRAY

An arrangement of elements in one or more dimensions. An M array is a set of nodes referenced by subscripts that share the same variable name.

ASCII

American **S**tandard **C**ode for **I**nformation **I**nterchange. A standardized coding scheme that assigns numeric values to letters, numbers, punctuation marks, and other characters to enable computer systems to exchange information.

AT-SIGN ("**@**")

A VA FileMan security Access code that gives the user programmer-level access to files and to VA FileMan's developer features. See Programmer Access. Also, the character "**@**" (i.e., at-sign, Shift-2 key on most keyboards) is used at VA FileMan field prompts to delete data.

BOOLEAN EXPRESSION

A logical comparison between values yielding a true or false result. In M, zero means false and non-zero (often one) means true.

BULLETIN

Electronic mail messages that are automatically delivered by MailMan under certain conditions. For example, a bulletin can be set up to fire when database changes occur, such as adding a record to the file of users. Bulletins are fired by bulletin-type cross-references.

CALLABLE ENTRY POINT

An authorized programmer call that may be used in any **V**_IST_**A** application package. The DBA maintains the list of DBIC-approved entry points.

CARET

A symbol expressed as up caret ("^"), left caret ("<"), or right caret (">"). In many M systems, a right caret is used as a system prompt and an up caret as an exiting tool from an option. Also known as the up-arrow symbol or shift–6 key.

CHECKSUM

The result of a mathematical computation involving the individual characters of a routine or file.

CLIENT

A single term used interchangeably to refer to the user, the workstation, and the portion of the program that runs on the workstation. In an object-oriented environment, a client is a member of a group that uses the services of an unrelated group. If the client is on a local area network (LAN), it can share resources with another computer (server).

COMPONENT

An object-oriented term used to describe the building blocks of GUI applications. A software object that contains data and code. A component may or may not be visible. These components interact with other components on a form to create the GUI user application interface.

COTS

Commercial **O**ff-**t**he-**S**helf. COTS refers to software packages that can be purchased by the public and used in support of **V**_IST_**A**.

CROSS-REFERENCE

An attribute of a field or a file that identifies an action that should take place when the value of a field is changed. Often, the action is the placement of the field's value into an index. A Traditional cross-reference is defined with a specific field. A New-Style cross-reference is a file attribute and can be composed of one or more fields. New-Style cross-references are stored in the INDEX file (#.11).

A cross-reference on a file provides direct access to the entries in several ways. For example, the PATIENT file is cross-referenced by name, Social Security Number (SSN), and bed number. When asked for a patient, the user can then respond with the patient's name, SSN, or bed number. Using cross-references speeds up access to the file for printing reports. A cross-reference is also referred to as an index or cross-index.

DATA DICTIONARY

The Data Dictionary is a global containing a description of the kind of data that is stored in the global corresponding to a particular file. VA FileMan uses the data internally for interpreting and processing files.

A Data Dictionary (DD) contains the definitions of a file’s elements (fields or data attributes), relationships to other files, and structure or design. Users generally review the definitions of a file's elements or data attributes; programmers review the definitions of a file's internal structure.

DATA DICTIONARY ACCESS

A user's authorization to write, update, and/or edit the data definition for a computer file (also known as DD Access).

DBA

Data**b**ase **A**dministrator.

DBIA

Data**b**ase **I**ntegration **A**greement, a formal understanding between two or more application packages that describes how data is shared or how packages interact. The DBA maintains a list of DBIAs between package developers, allowing the use of internal entry points or other package-specific features that are not available to the general programming public.

DEFAULT

A response the computer considers the most probable answer to the prompt being given. In the roll-and-scroll mode of **V***IST***A**, the default value is identified by double forward slash marks (//) immediately following it. In a GUI-based application the default may be a highlighted button or text. This allows you the option of accepting the default answer or entering your own answer. To accept the default you simply press the enter (or return) key. To change the default answer, type in your response.

DIRECT MODE UTILITY

A programmer call that is made when working in direct programmer mode. A direct mode utility is entered at the M prompt (e.g., >**D ^XUP**). Calls that are documented as direct mode utilities *cannot* be used in application package code.

DLL

Dynamic **L**ink **L**ibrary. A DLL allows executable routines to be stored separately as files with a DLL extension. These routines are only loaded when a program calls for them. DLLs provide several advantages:

1. DLLs help save on computer memory, since memory is only consumed when a DLL is loaded. They also save disk space. With static libraries, your application absorbs all the library code into your application so the size of your application is greater. Other applications using the same library will also carry this code around. With the DLL, you don’t carry the code itself, you have a pointer to the common library. All applications using it will then share one image.

2. DLLs ease maintenance tasks. Because the DLL is a separate file, any modifications made to the DLL will not affect the operation of the calling program or any other DLL.

3. DLLs help avoid redundant routines. They provide generic functions that can be utilized by a variety of programs.

DOMAIN

A site for sending and receiving mail.

DOUBLE QUOTE (")

A symbol used in front of a Common option's menu text or synonym to select it from the Common menu. For example, the four-character string "TBOX" selects the User's Toolbox Common option.

ERROR TRAP

A mechanism to capture system errors and record facts about the computing context such as the local symbol table, last global reference, and routine in use. Operating systems provide tools such as the %ER utility. The Kernel provides a generic error trapping mechanism with use of the ^%ZTER global and ^XTER* routines. Errors can be trapped and, when possible, the user is returned to the menu system.

FIELD

In an entry, a specified area used to hold values. The specifications of each VA FileMan field are documented in the file's data dictionary.

FIELD NUMBER

The unique number used to identify a field in a file. A field can be referenced by "#" followed by the field number.

FORUM

The central e-mail system within **V***IST***A**. Developers use FORUM to communicate at a national level about programming and other issues. FORUM is located at the Washington, DC CIO Field Office (162-2).

FREE TEXT

The use of any combination of numbers, letters, and symbols when entering data.

GUI

Graphical **U**ser Interface. A type of display format that enables users to choose commands, initiate programs, and other options by selecting pictorial representations (icons) via a mouse or a keyboard.

HELP FRAMES

Entries in the HELP FRAME file (#9.2) that can be distributed with application packages to provide online documentation. Frames can be linked with other related frames to form a nested structure.

HELP PROMPT

The brief Help that is available at the field level when entering one or more question marks.

ICON

A picture or symbol that graphically represents an object or a concept.

IDCU

The **I**ntegrated **D**ata **C**ommunications **U**tility that is a wide area network (WAN) used by VA for transmitting data between VA sites.

INITIALIZATION

The process of setting variables in a program to their starting value.

INPUT TRANSFORM

An executable string of M code that is used to check the validity of input and converts it into an internal form for storage.

INTERNAL ENTRY NUMBER

The number used to identify an entry within a file. Every record has a unique internal entry number. Often abbreviated as IEN.

IRM

Information **R**esource **M**anagement. A service at VA medical centers responsible for computer management and system security.

KERNEL

A set of **V**_IST_**A** software routines that function as an intermediary between the host operating system and the **V**_IST_**A** application packages (e.g., Laboratory, Pharmacy, IFCAP, etc.). Kernel provides a standard and consistent user and programmer interface between application packages and the underlying M implementation. (VA FileMan and MailMan are self-contained to the extent that they can standalone as verified packages.) Some of Kernel's components are listed below along with their associated namespace assignments:

KIDS	XPD
Menu Management	XQ
Tools	XT
Sign-on/Security	XU
Device Handling	ZIS
Task Management	ZTM

KEY

The purpose of Security Keys is to set a layer of protection on the range of computing capabilities available with a particular software package. The availability of options is based on the level of system access granted to each user.

KEYWORD

A reference name that calls a Help Frame when entered at a message prompt.

KIDS

Kernel **I**nstallation & **D**istribution **S**ystem.

LOCAL

The system to which a user is currently signed on.

LOOKUP

To find an entry in a file using a value for one of its fields.

MAILMAN

The Kernel module that provides a mechanism for handling electronic communication, whether it is user-oriented mail messages, automatic firing of bulletins, or initiation of server-handled data transmissions.

MENU MANAGER

The Kernel module that controls the presentation of user activities such as menu choices or options. Information about each user's menu choices is stored in the Compiled Menu System, the ^XUTL global, for easy and efficient access.

MESSAGE-ID

A message identifier that shows the message number and the domain name of the message.

METHOD

An object-oriented term used to describe procedures and functions, also referred to as routines, associated with a particular component. Methods are called at run-time. They are never activated at design-time.

MULTIPLE

A multiple-valued field; a subfile. In many respects, a multiple is structured like a file.

MUMPS (ANSI STANDARD)

A programming language recognized by the **A**merican **N**ational **S**tandards **I**nstitute (ANSI). The acronym MUMPS stands for **M**assachusetts General Hospital **U**tility **M**ulti-**p**rogramming **S**ystem and is abbreviated as M.

NAME FIELD

The one field that must be present for every file and file entry. It is also called the .01 field. At a file's creation, the .01 field is given the label NAME. This label can be changed.

NAMESPACING

A convention for naming **V***IST***A** package elements. The Database Administrator (DBA) assigns unique character strings for package developers to use in naming routines, options, and other package elements so that packages may coexist. The DBA also assigns a separate range of file numbers to each package.

NODE

In a tree structure, a point at which subordinate items of data originate. An M array element is characterized by a name and a unique subscript. Thus the terms: node, array element, and subscripted variable are synonymous. In a global array, each node might have specific fields or "pieces" reserved for data attributes such as name.

NULL

Empty. A field or variable that has no value associated with it is null.

ON THE FLY

A term given to the process of not permanently storing data in the data dictionary but having a computation performed at run time.

OPTION

As an item on a menu, an option provides an opportunity for users to select it, thereby invoking the associated computing activity. In **V***IST***A**, an entry in the OPTION file (#19). Options may also be scheduled to run in the background, non-interactively, by TaskMan.

PATTERN MATCH

In M, an operator that compares the contents of a variable or literal to a specified pattern of characters or kinds of characters.

PERIPHERAL DEVICE

Any hardware device other than the computer itself (central processing unit plus internal memory). Typical examples include card readers, printers, CRT units, and disk drives.

POINTER

Points to another file where the computer stores information needed for the field of the file in which you are currently working. If you change any of the information in the field in which you are working, the new information is automatically entered into the "pointed to" file.

PROGRAMMER ACCESS

The ability to use VA FileMan features that are reserved for application developers. Referred to as "having the at-sign ('@')" because the at-sign is the DUZ(0) value that grants programmer access.

PROMPT

The computer interacts with the user by issuing questions called *prompts*, to which the user returns a response.

PROPERTIES

An object-oriented term used to describe the attributes associated with components on a GUI form. These attributes, collectively, indicate how a component is to be displayed to the user in the application interface. Properties are activated at design time.

QUEUING

Requesting that a job be processed in the background rather than in the foreground within the current session. Jobs are processed sequentially (first-in, first-out). Kernel's TaskMan module handles the queuing of tasks.

READ ACCESS

A user's authorization to read information stored in a computer file.

REMOTE PROCEDURE CALL

A remote procedure call (RPC) is essentially M code that may take optional parameters to do some work and then return either a single value or an array back to the client application.

ROUTINE

A program or a sequence of instructions called by a program that may have some general or frequent use. M routines are groups of program lines that are saved, loaded, and called as a single unit via a specific name.

SAC

Standards **a**nd **C**onventions. Through a process of verification, **V***IST***A** packages are reviewed with respect to SAC guidelines as set forth by the Standards and Conventions Committee (SACC). Package documentation is similarly reviewed in terms of standards set by the Documentation Standards and Conventions Committee (DSCC).

SACC

V*IST***A** **S**tandards **a**nd **C**onventions **C**ommittee. This Committee is responsible for maintaining the document called SAC.

SECURITY KEY

The purpose of Security Keys is to set a layer of protection on the range of computing capabilities available with a particular software package. The availability of options is based on the level of system access granted to each user.

SERVER

The computer where the data and the Business Rules reside. It makes resources available to client workstations on the network. In **V*IST*A**, it is an entry in the OPTION file (#19). An automated mail protocol that is activated by sending a message to a server at another location with the "S.server" syntax. A server's activity is specified in the OPTION file (#19) and can be the running of a routine or the placement of data into a file.

SIGN-ON/SECURITY

The Kernel module that regulates access to the menu system. It performs a number of checks to determine whether access can be permitted at a particular time. A log of signons is maintained.

SUBSCRIPT

A symbol that is associated with the name of a set to identify a particular subset or element. In M, a numeric or string value that: is enclosed in parentheses, is appended to the name of a local or global variable, and identifies a specific node within an array.

SYSTEM MANAGER/IRM CHIEF

At each site, the individual who is responsible for managing computer systems, installing and maintaining new modules, and serving as liaison to the CIOFOs.

TASKMAN

The Kernel module that schedules and processes background tasks (also called Task Manager).

TRIGGER

A trigger is an instruction that initiates a procedure. In VA FileMan, a trigger can be set up when entry of data in one field automatically updates a second field value.

UCI

User **C**lass **I**dentification, a computing area. The MGR UCI is typically the Manager's account, while VAH or ROU may be Production accounts.

USER ACCESS

This term is used to refer to a limited level of access to a computer system that is sufficient for using/operating a package, but does not allow programming, modification to data dictionaries, or other operations that require programmer access. Any of **V*IST*A**'s options can be locked with a security key (e.g., XUPROGMODE, which means that invoking that option requires programmer access).

The user's access level determines the degree of computer use and the types of computer programs available. The Systems Manager assigns the user an access level.

USER INTERFACE

The way the package is presented to the user, such as Graphical User Interfaces that display option prompts, help messages, and menu choices. A standard user interface can be achieved by using Borland's Delphi Graphical User Interface to display the various menu option choices, commands, etc.

VA FILEMAN

(ALSO CALLED VA FILEMANAGER)

A set of programs used to enter, maintain, access, and manipulate a database management system consisting of files. A package of online computer routines written in the M language which can be used as a stand-alone database system or as a set of application utilities. In either form, such routines can be used to define, enter, edit, and retrieve information from a set of computer-stored files.

VARIABLE

A symbol representing a value that changes during the execution of a routine.

VERIFY CODE

The Kernel's Sign-on/Security system uses the Verify code to validate the user's identity. This is an additional security precaution used in conjunction with the Access code. Verify codes shall be at least eight characters in length and contain three of the following four kinds of characters: letters (lower- and uppercase), numbers, and, characters that are neither letters nor numbers (e.g., "**#**", "**@**" or "**$**"). If entered incorrectly, the system does not allow the user to access the computer. To protect the user, both codes are invisible on the terminal screen.

VISTA

Veterans Health **I**nformation **S**ystems and **T**echnology **A**rchitecture. **V***IST***A** includes the VA's application software (i.e., Microsoft Windows-based and locally-developed applications, roll-and-scroll, and interfaces such as software links to commercial packages). In addition, it encompasses the VA's uses of new automated technology including the clinical workstations. **V***IST***A** encompasses the rich automated environment already present at local VA medical facilities.

WINDOW

An object on the screen (dialog) that presents information such as a document or message.

Appendix A—Patch Revision History

The following table displays the patch/version release history for the RPC Broker software. The sequence number (Seq #) is the order in which the patch was released by National **V*IST*A** Support (NVS) and installed by the site. The sequence number does not necessarily match the Patch ID number in all cases. Also, the sequence number, in some cases, can imply dependency between patches. Each table entry indicates that the documentation was reviewed and updated as needed for each patch; in some cases, a patch may not affect the content of the documentation. Regardless, the patch will still be added to the patch history in reverse patch sequence order.

Seq #	Patch ID	Brief Summary	Status
22	XWB*1.1*26	This patch updates the Broker's Programmer Client Workstation software—also known as the Broker Development Kit (BDK). It supports Delphi V. 4, 5, and 6. It provides a SharedRPCBroker component. Any GUI application that uses the SharedRPCBroker will now have the ability to share a Broker connection. This patch also supports ESSO.	Client-side only patch—Not yet released. This document has been reviewed and updated as needed for this patch.
21	XWB*1.1*13	This patch updates the Broker's Programmer Client Workstation software—also known as the Broker Development Kit (BDK). It supports Delphi V. 4, 5, and 6. It provides Silent Login functionality in the Broker. Any GUI RPC Broker-based application will now have the ability to login to an M Server silently (i.e. without any user dialog). This patch also supports Enterprise Single-Sign-On (ESSO).	Client and server patch—Not yet released. This document has been reviewed and updated as needed for this patch.
20	XWB*1.1*27	This patch enables asynchronous processing, multiple jobs running at the same time. Prior to this patch, processing of requests to the HL7 package for remote data made by GCPR and CPRS, was performed synchronously - in order of time of request, each job finishing before the next job started.	Server-side only patch—Patch released on 03/15/02.
19	XWB*1.1*16	This patch provides several bug fixes (e.g., READ/WRITE errors) initiated via NOIS.	Server-side only patch—Patch released on 02/06/02.

Seq #	Patch ID	Brief Summary	Status
18	XWB*1.1*24	This patch updates the Broker's Programmer Client Workstation software—also known as the Broker Development Kit (BDK). It supports only Delphi V. 4 and Delphi V. 5. Due to version-dependent code, a problem was recently encountered that is associated with reading the Microsoft Windows Registry in programs compiled with Delphi V. 5. Because a conditional test was specifically looking for Delphi V. 4-based applications, Delphi V. 5-based applications ended up using Broker code for Delphi V. 3. This can result in users having limited privileges, preventing their ability to read data from the registry. This has been observed when a user with limited NT privileges attempts to select a location for the RPC Broker connection, and it results in the use of the default BrokerServer/9200. However, users with higher levels of NT access do not see this problem. This version-dependent code was removed via this patch.	Client and server patch—Patch released on 11/09/01. This document was reviewed and updated as needed for this patch.
17	XWB*1.1*22	The calling site had a NEW PERSON file entry with a phone number containing a trailing backslash ("\"). As part of Remote Data Views (RDV), this data was then encoded and sent to the remote site. At the remote site, a bug caused the backslash ("\") to be appended to the end of several other strings, which then caused the reported error. This was fixed by correcting the decoding routine. Because the error occurred before RDV was setup to handle an error, it caused the calling site to keep sending the same message repeatedly. This has been fixed by setting an error trap at the beginning of RDV. If the application does not set some data into the return variable, XWB2HL7 will return a string starting with "-1^". The XWB EXAMPLE option, RPC's and routine (XWBEXMPL) are included to add an entry point for testing that will record the symbol table in the error trap.	Server-side only patch—Patch released on 10/03/01. This document was reviewed and updated as needed for this patch.

Seq #	Patch ID	Brief Summary	Status
16	XWB*1.1*20	This patch addresses the following: - During the early testing of RDV (Remote Data View), the DUZ value was hard set to .5 just before the call to the RPC. This was done because the code to set up the user at the remote site wasn't ready. When the code was fixed to properly set the DUZ, the old code was never removed. This has been fixed in the routine XWB2HL7. - If data was left in the ^XUTL("XQ",$J,"IO")node it could cause problems when HOME^%ZIS is called by some RPC's, so this ^XUTL node is killed off before the RPC is called. - In an e-mail message from CPRS developers: The global that may be used to pass data back to the RPC was not killed before its use. This was fixed in the routine XWBDRPC.	Server-side only patch—Patch released on 05/10/01. This document was reviewed and updated as needed for this patch.
15	XWB*1.1*14	This patch updates the Broker's Programmer Client Workstation software—also known as the Broker Development Kit (BDK). It adds no new functionality . It does the following: - Releases the source code for the BDK. - Splits the VistaBroker package into separate design- and run-time packages.	Client and server patch—Patch released on 10/17/00. This document was reviewed and updated as needed for this patch.
14	XWB*1.1*18	This patch fixed the following NOIS: LOM-0800-62301 and PRO-0800-11778: If there are problems associated with the remote site's HL7 definitions—specifically the receiving application. Then the RPC XWB REMOTE STATUS CHECK will get an UNDEF error on the variable Z.	Server-side only patch—Patch released on 10/17/00. This document was reviewed and updated as needed for this patch.
13	XWB*1.1*12	This patch is in support of the CPRS Remote Data Views project. The RPC Broker is used to facilitate invocation of Remote Procedure calls on a remote server. The RPC Broker uses **V***IST***A** HL7 as the vehicle to pass RPC name and parameters from a local server to a remote server. On the return path, **V***IST***A** HL7 is also used to send results from the remote server back to the local server.	Server-side only patch—Patch released on 08/04/00. This document was reviewed and updated as needed for this patch.

Seq #	Patch ID	Brief Summary	Status
12	XWB*1.1*10	This patch gives greater information about and control of RPCs. Specific new abilities are: • Blocking an RPC either locally*, remotely*, or in both contexts by setting a value in the INACTIVE field of the Remote Procedure file. Prior to this patch, values in this field had no effect. • Assuring that an RPC is at least a specified version when it is run remotely* by setting a value in the new VERSION field of the REMOTE PROCEDURE file. • Querying a server regarding the status of RPCs by using new Remote Procedures: XWB IS RPC AVAILABLE and XWB ARE RPCS AVAILABLE. • In addition, this patch stops M errors from occurring when a client application attempts to: 1.) Create a context that does not exist on the server, or 2.) Run a remote procedure that does not exist on the server.	Server-side only patch—Patch released on 08/04/00. This document was reviewed and updated as needed for this patch.
11	XWB*1.1*15	This patch should correct a problem on Cache sites with the Broker looping with COMMAND errors. This error is caused when the Broker tries to open the TCP port and the port is already open via the Broker.	Server-side only patch—Patch released on 04/12/00. This document was reviewed and updated as needed for this patch.
10	XWB*1.1*11	This patch updates the Broker's Programmer Client Workstation software—also known as the Broker Development Kit (BDK)—adding support for Delphi V. 5 development.	Client and server patch—Patch released on 01/24/00. This document was reviewed and updated as needed for this patch.
9	XWB*1.1*9	This patch fixes the following: • Intersystems License. This is the patch that works with Patch XU*8*118. The code to share licenses when GUI and Telnet users from the same workstation are connected is in place and ZU now calls it. This patch adds a similar call from XWBTCPC. • This patch brings a new XWB LISTENER STOP ALL option for shutting down multiple listeners. It also brings a modified option XWB LISTENER STARTER for starting Broker listeners.	Server-side only patch—Patch released on 01/24/00. This document was reviewed and updated as needed for this patch.

Seq #	Patch ID	Brief Summary	Status
8	XWB*1.1*8	This patch supports GUI Multi-Divisional signon. If a user has more than one division to choose from, the user must select one before continuing with the signon. If the user has only one division in File #200, this division will be used; otherwise, the default institution in the KERNEL SYSTEM PARAMETERS file will be used.	Client-side only patch—Patch released on 12/10/99. This document was reviewed and updated as needed for this patch.
7	XWB*1.1*6	This patch does the following: • Eliminates server Broker jobs for which there is no client application. • Changes the time that the server waits for the client to contact it. A new field in the KERNEL SYSTEM PARAMETERS file, BROKER ACTIVITY TIMEOUT (default value of approximately 3 minutes) controls the length of the timeout.	Client and server patch—Patch released on 09/09/99. This document was reviewed and updated as needed for this patch.
6	XWB*1.1*4	This patch does the following: 1. Introduces a shorter timeout when logging in via any GUI RPC Broker-based application. The server listener process will timeout after 90 seconds if the user has not passed in his/her Access and Verify codes. 2. Updates the Broker's Programmer Client Workstation software—also known as the Broker Development Kit (BDK)—adding support for Delphi V. 4 development. 3. Fixes a bug in which the Title bar of the Kernel Login form was being changed when a user started entering their Access code.	Client and server patch—Patch released on 06/24/99. This document was reviewed and updated as needed for this patch.
5	XWB*1.1*7	This patch addresses two problems: 1. A command error is occurring at RESTART+17^XWBTCPL when the Broker tries to reopen a device that is not closed. This seems to be a problem with Cache sites only. The result of this error causes the Broker Listener to stop. The fix is in XWBTCPL. 2. The listener doesn't check for available slots before starting a new process. The listener will now check the MAX SIGNON ALLOWED field of the VOLUME SET multiple in the KERNEL SYSTEM PARAMETERS file, the same one used by Kernel logon. This fix is also in XWBTCPL.	Server-side only patch—Patch released on 06/04/99. This document was reviewed and updated as needed for this patch.

Seq #	Patch ID	Brief Summary	Status
4	XWB*1.1*5	This patch is for the support of RUM. This will allow the trapping of data for Remote Procedure Calls (RPCs) and the RPC Broker handler.	Server-side only patch—Patch released on 03/31/99. This document was reviewed and updated as needed for this patch.
3	XWB*1.1*3	Under CPRS, when the DG routines call OP^XQCHK to record what option is used, it was getting back "unknown." The Broker created context needed to set the variable XQY.	Server-side only patch—Patch released on 01/06/99. This document was reviewed and updated as needed for this patch.
2	XWB*1.1*2	This patch addresses three problems with RPC Broker v1.1: • Encrypted Literal—Pattern match failure in RPCs. The failure only occurs with RPCs that combine multiple literals and an array (NOIS WAS-0398-22800). • Data Collection Switch turned "Off"—Collection of data will be controlled by the use of the Capacity Management tools (NOIS BRX-0498-11768 and HUN-0498-21137). • 10 Second Network Timeout in Client Agent—A 30 second timeout is being switched to 10 for network communications with the Client Agent.	Server-side only patch—Patch released on 07/27/98. This document was reviewed and updated as needed for this patch.
1	XWB*1.1*1	This patch fixes some small problems that were discovered after release (server-side only). • XWBTCPL—Remove the SYMBOL_TABLE from the VAX DSM JOB command. • XWBTCP—When stopping the Broker, see a failure to open a socket. • XWB BROKER EXAMPLE option—This option was missing its type field.	Server-side only patch—Patch released on 02/18/98. This document was reviewed and updated as needed for this patch.
NA	Version 1.1	Original Version 1.1 software release.	September 1997

Table 8: RPC Broker V. 1.1 patch revision history (in reverse sequence order)

Index